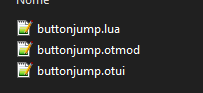
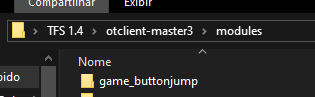
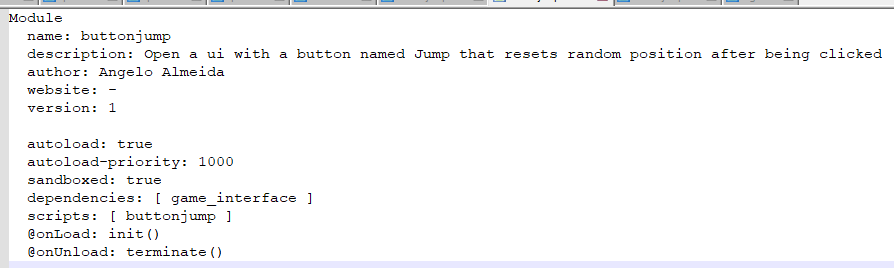
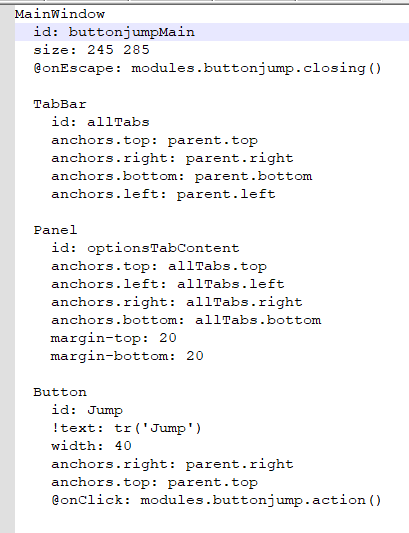
I found very little informations and documentations of the modules of OTC, so I invested a loto f time testing and discovering by myself, trying to get a little parts of other modules that could fit my needs. So I started by creating the module of the window with the button.



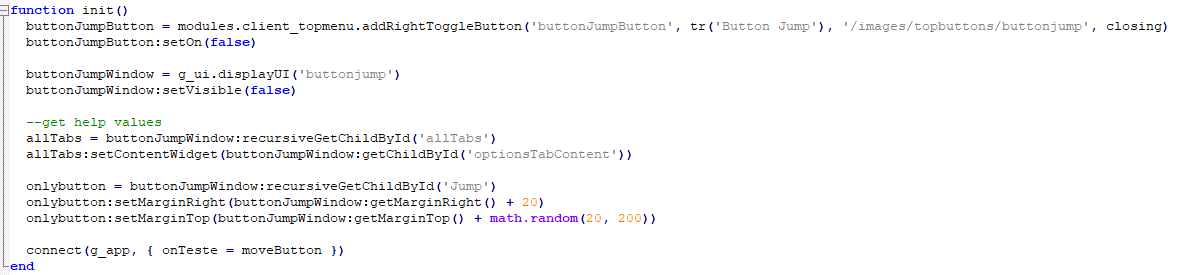
Starting with the .otmod, which is the simplier, I just added information and some values necessary to create the module, as the following image shows. It’s pretty standard for all modules.



Then, at the .otui I had to specify the size of the window, set some anchors and margins for the button to be drawn inside. The ids are very importante, as we use them to get the function child.



Finally at the .lua file I have to declare some essential functions, as init() and terminate(), as well as functions to move the button and a function to handle what happens when you click the button.



I use buttonJumpButton = modules.client\_topmenu.addRightToggleButton and pass the information to create a button on the OTC UI, so I can open the window.

The only way I found to make the button move arround the window, was to manipulate the margins of the button, incrementing it and setting the Y to a random between 20 and 200, but i’m eager to know if there’s a better way. (I tried setX, setPosition, move, lots of functions and none of them worked). I do this by getting the button calling for the children by Id, then I manipulate it’s margin.

onlybutton = buttonJumpWindow:recursiveGetChildById('Jump')

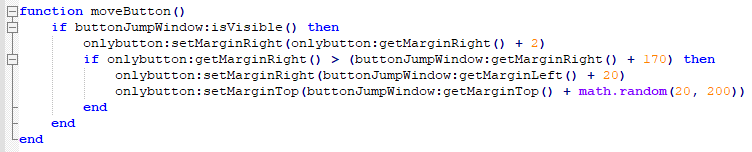
onlybutton:setMarginRight(buttonJumpWindow:getMarginRight() + 20)

onlybutton:setMarginTop(buttonJumpWindow:getMarginTop() + math.random(20, 200))

The more importante line of code is the connect, as I bind it to a event of the OTC, called onTeste (sorry I could’ve named it better but I was out of time). I do this to update the button after 100ms has elapsed, I tried to use other call-backs but they were too slow or too fast (1000 ms or 4000 microseconds for example).

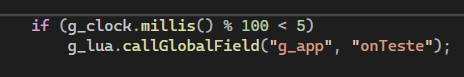
connect(g\_app, { onTeste = moveButton })

The terminate and closing functions are pretty standard and self-explanatory, so the next one to comment is the moveButton, this function manipulates the margins, pushing the button 2 units at a time to the left, and when it gets more than 170 units away from the beginning margin, it resets and I set it’s Y randomly.



The last thing is the click, I called it action() and it checks if the window is visible, if it is, he resetes the left margin to the beginning one and resets the Y to a random between 20 and 200 like the previous.

Now on the code of the OTC I just added the luacall onTeste on the graphicalapplication.cpp, since I was using onFps before and it’s located at this cpp.



And just like this I made the button move. This was a tricky question, that seemed to be easy at first but I spent almost 3 days discovering the methods and logic to be able to reproduce the reference from the video. I learned a lot implementing this and I’d love to see how it was supposed to work, if implemented by someone who knows this deeply.

PS: I only commented on the .lua file because comments on the otui and otmod would make them not load, so I hope it’s ok if I explained them here.